

Chapter 8 - Protective Measures

SI539 - Charles Severance

Textbook: Build Your own Ruby on Rails Application by Patrick Lenz (ISBN:978-0-975-8419-5-2)

Welcome to SI539

- About
- Contact
- Pictures
- Membership
- Chat
- Application

Please Log In

Required Information

Enter your E-Mail:

Enter your Password:

If you have lost your password,
membership@si539.com to have



AUTHENTICATION REQUIRED::

You are connecting to a U-M website that requires authentication. Please enter your Login ID (username or Friend ID) and password to continue.

Need a Login ID?

If you don't have a Login ID, you can [create one now](#).

Login ID	<input type="text"/>
Password	<input type="password"/>
<input type="button" value="▶ MToken"/>	
<input type="button" value="Log In"/>	
Forgot your password?	
Login Help	

By using this service you agree to adhere to [U-M computing policies and guidelines](#).

Some Web sites always seem to want to know who you are!


twitter

Home Find & Follow Public Timeline Settings Help Sign out

What are you doing?

140

Hi, [your profile](#)

 drchuck

update

Recent

Replies

Archive



dkeats Spending the morning with the eLearning team innovation, and participation in a community of practice Energizing. 3 minutes ago from web ☆ ↻



drchuck Working on lecture slides - talking about cool sessions. 5 minutes ago from web ☆ ↻



Matth Ordered Urban Fortunes from Amazon because saying it said things I wasn't at all agreeing with. Right. ago from [twitterrific](#) ☆ ↻



microcline Back from walking the dogs. Cleaned them can, but they don't get to share the futon tonight. 31 m from web ☆ ↻

flickr

Home You Organize Contacts Groups Explore

Signed in as dr-chuck (1 new)

Search everyone's photos



Ni hao dr-chuck!

Now you know how to greet people in Mandarin!

- You have 1 new message.
- [Find your friends](#)

Flickr News

31 Mar 08 - Flickr is more fun with friends, but it is a big busy place, and sometimes it can be hard to find the people you know. The new Find Your... [read more news](#)

» [Flickr Blog](#) Great photos & latest news, daily!

ADVERTISEMENT



» [Upload Photos](#) (Or, look at our uploading [tools](#)...)

» [Your Photos](#) ([Recent activity](#) / [Comments you've made](#))



» [Photos from your Contacts](#)



From [Lance70](#)



From [Lance70](#)



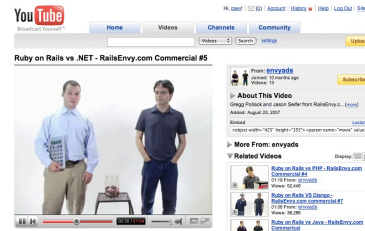
From [Lance70](#)



From [Lance70](#)

Other Web sites always seem to know who you are!

Browser



Click Draw

Whole
Page

GET

You watch the YouTube video
for an 30 seconds

Click Draw

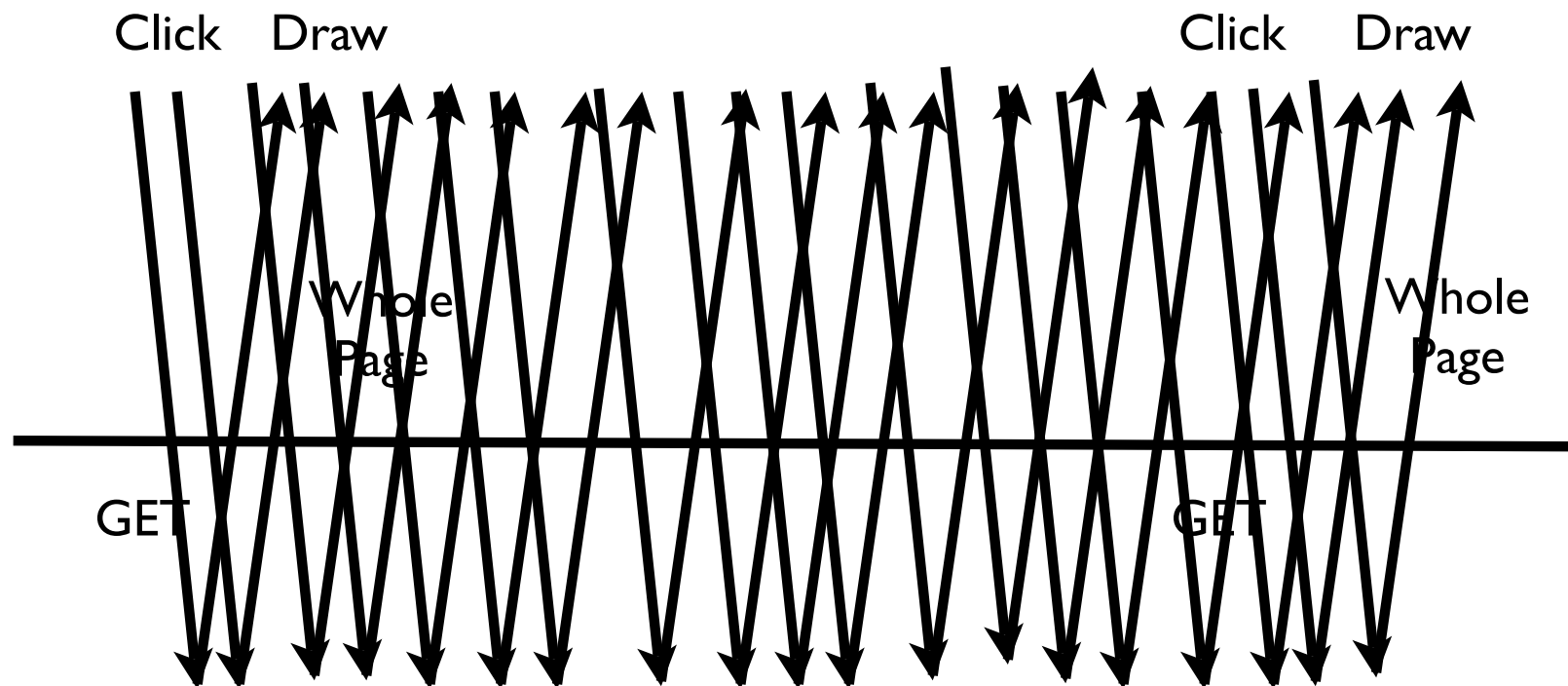
Whole
Page

GET

Server

How you see YouTube...

Browser



Server

How YouTube sees you...

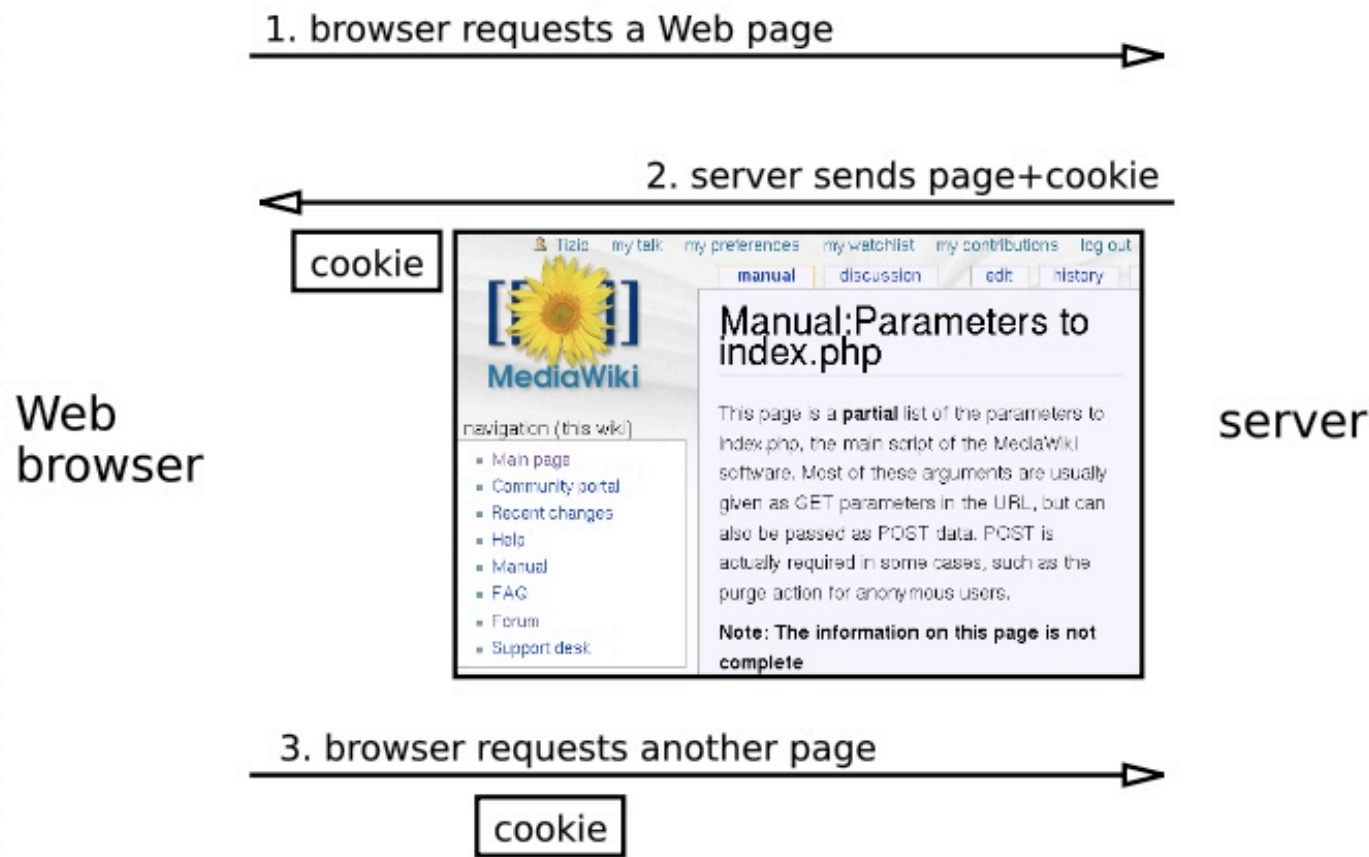
Multi-User

- When a server is interacting with many different browsers at the same time, the server needs to know *which* browser a particular request came from
- Request / Response initially was stateless - all browsers looked identical - this was really really bad and did not last very long at all.

Web Cookies to the Rescue

Technically, cookies are arbitrary pieces of data chosen by the Web server and sent to the browser. The browser returns them unchanged to the server, introducing a state (memory of previous events) into otherwise stateless HTTP transactions. Without cookies, each retrieval of a Web page or component of a Web page is an isolated event, mostly unrelated to all other views of the pages of the same site.

http://en.wikipedia.org/wiki/HTTP_cookie



http://en.wikipedia.org/wiki/HTTP_cookie

Cookies In the Browser

- Cookies are marked as to the web addresses they come from - the browser only sends back cookies that were originally set by the same web server
- Cookies have an expiration date - some last for years - others are short-term and go away as soon as the browser is closed

Playing with Cookies

- Firefox Developer Plugin has a set of cookie features
- Other browsers have a way to view or change cookies

Welcome to SI539

- **About**
- **Contact**
- **Pictures**
- **Membership**
- **Chat**
- **Application**
- **Logout**

Welcome to the SI539 Sakai

Here are two quotes from Ghandi that look like they reach the same conclusion.

Power is of two kinds. One is obtained by the use of force, the other by acts of love. *Power* based on force is effective and permanent then the other.

I hope to demonstrate that the real source of authority by a few, but by the acquisition of authority, when abused.

Cookie Information - http://localhost:3000/One

Sakai dr-chuck.com NWA Chuck's Media iPhone Navigation I3k CT SI 182 Sakai Based Service : ...

Disable Cookies Clear Session Cookies Delete Domain Cookies Delete Path Cookies View Cookie Information Add Cookie...

http://localhost:3000/One

1 cookie

NAME	VALUE	HOST	PATH	SECURE	EXPIRES
_assn6_session_id	fa3e49c915aad8bd9af1f768b8081aa6	localhost	/	No	At End Of Session

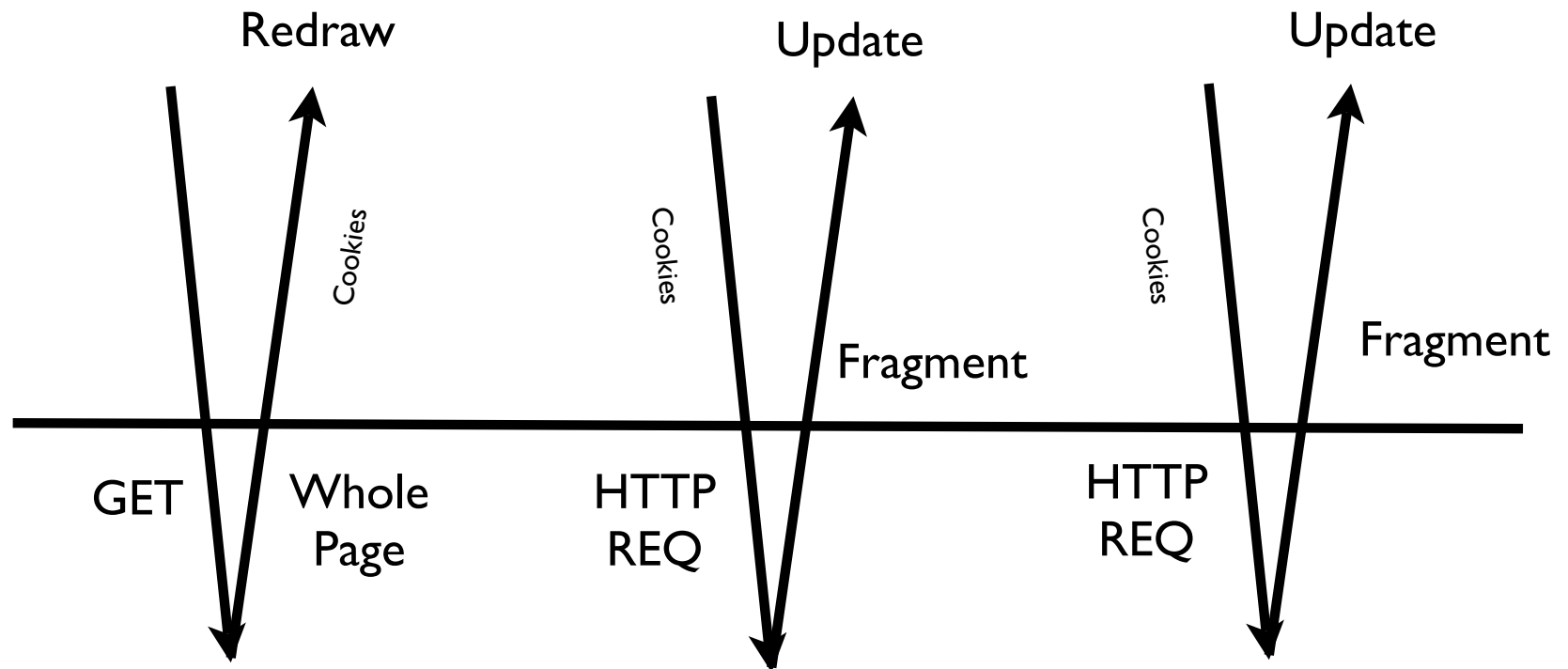
Done

Cookies

- Identifying Individual Users
- The Web is “stateless”
- How do we make the web seem not to be stateless

Request Response Again!

Browser



Server

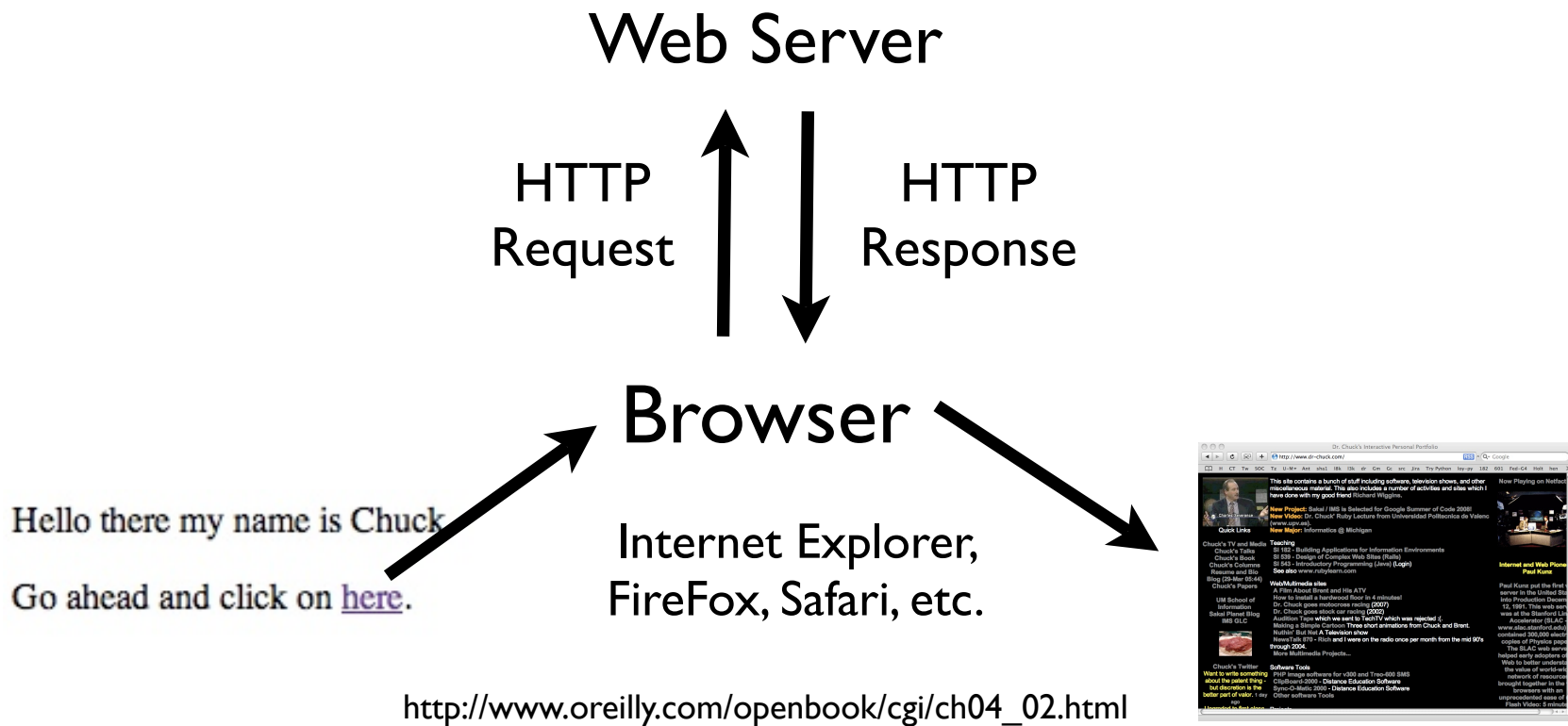
Remember that cookies are only sent back to the host that set the cookie.

<nerdy-stuff>

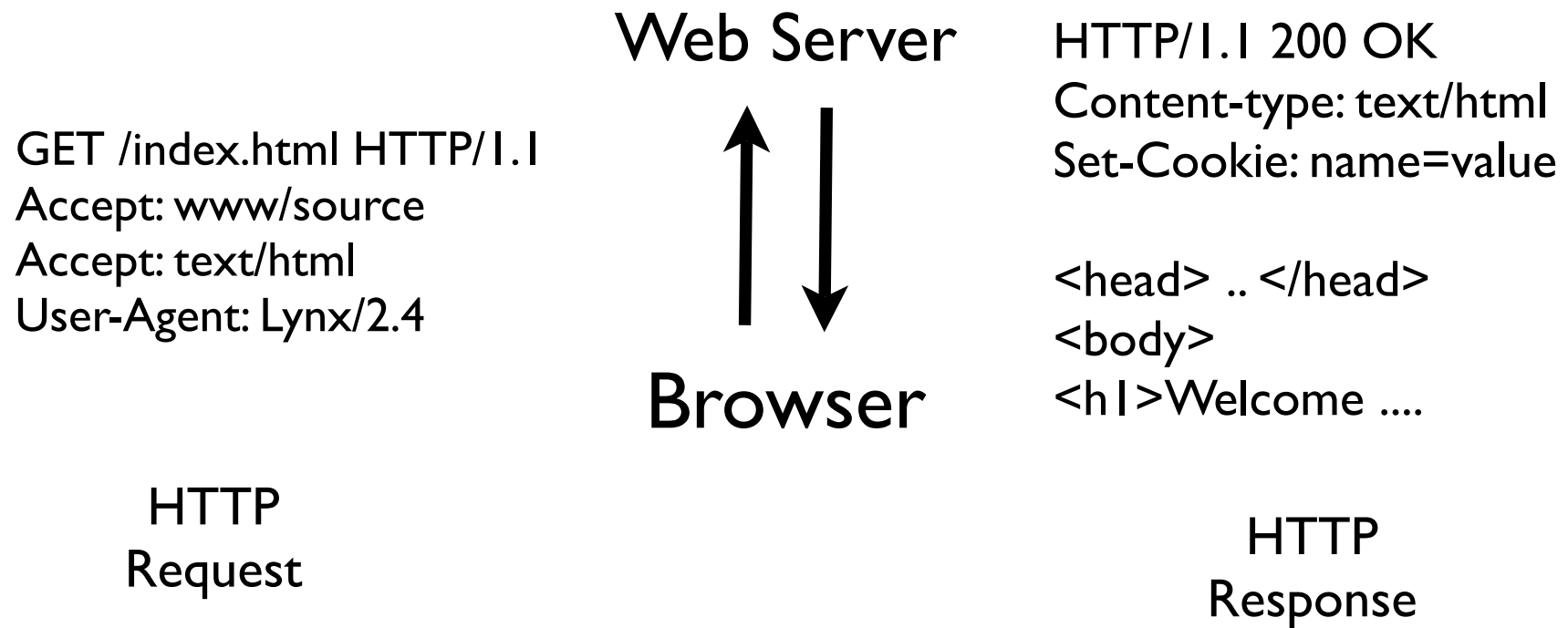
Getting Data From The Server

- Each time the user clicks on an anchor tag with an href= value to switch to a new page, the browser makes a connection to the web server and issues a “GET” request - to GET the content of the page at the specified URL
- The server returns the HTML document to the Browser which formats and displays the document to the user.

HTTP Request / Response Cycle



HTTP Request / Response Cycle



http://www.oreilly.com/openbook/cgi/ch04_02.html

HTTP Response / Request Cycle

HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=value

<head> .. </head>
<body>
<h1>Welcome

HTTP
Response

Web Server



Browser

GET /index.html HTTP/1.1
Accept: www/source
Accept: text/html
Cookie: name=value
User-Agent: Lynx/2.4

HTTP
Request

http://www.oreilly.com/openbook/cgi/ch04_02.html

`</nerdy-stuff>`

Sessions

- In Rails as soon as we meet a new browser - we create a session
- Rails sets a session cookie to be stored in the browser which indicates the session id in use
- The creation and destruction of sessions is generally transparent to Rails applications

Using Cookies to Support Sessions

Using Cookies Wisely

- Usually the server only stores a small amount of information in the cookie
 - Permanent - who you are - account name last access time
 - Temporary - session identifier

Session Identifier

- A large, random number that we place in a browser cookie the first time we encounter a browser.
- This number is used to pick from the many sessions that the server has active at any one time.
- Server software stores data in the session which it wants to have from one request to another from the same browser.
- Shopping cart or login information

Server

Browser A

cook=10

Browser B

cook=46

Browser C

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$500

withdraw:

bal=bal-100

Server

Browser A

cook=10

Browser B

cook=46

Browser C

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$500

withdraw:

bal=bal-100

→
Click

Server

Browser A

cook=10

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$500

Browser B

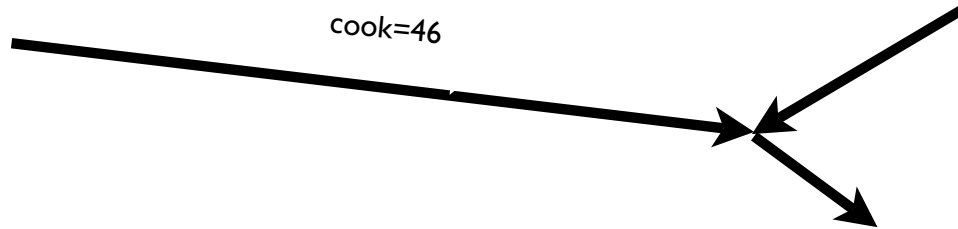
cook=46

cook=46

Browser C

withdraw:

bal=bal-100



Server

Browser A

cook=10

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

Request

Browser B

cook=46

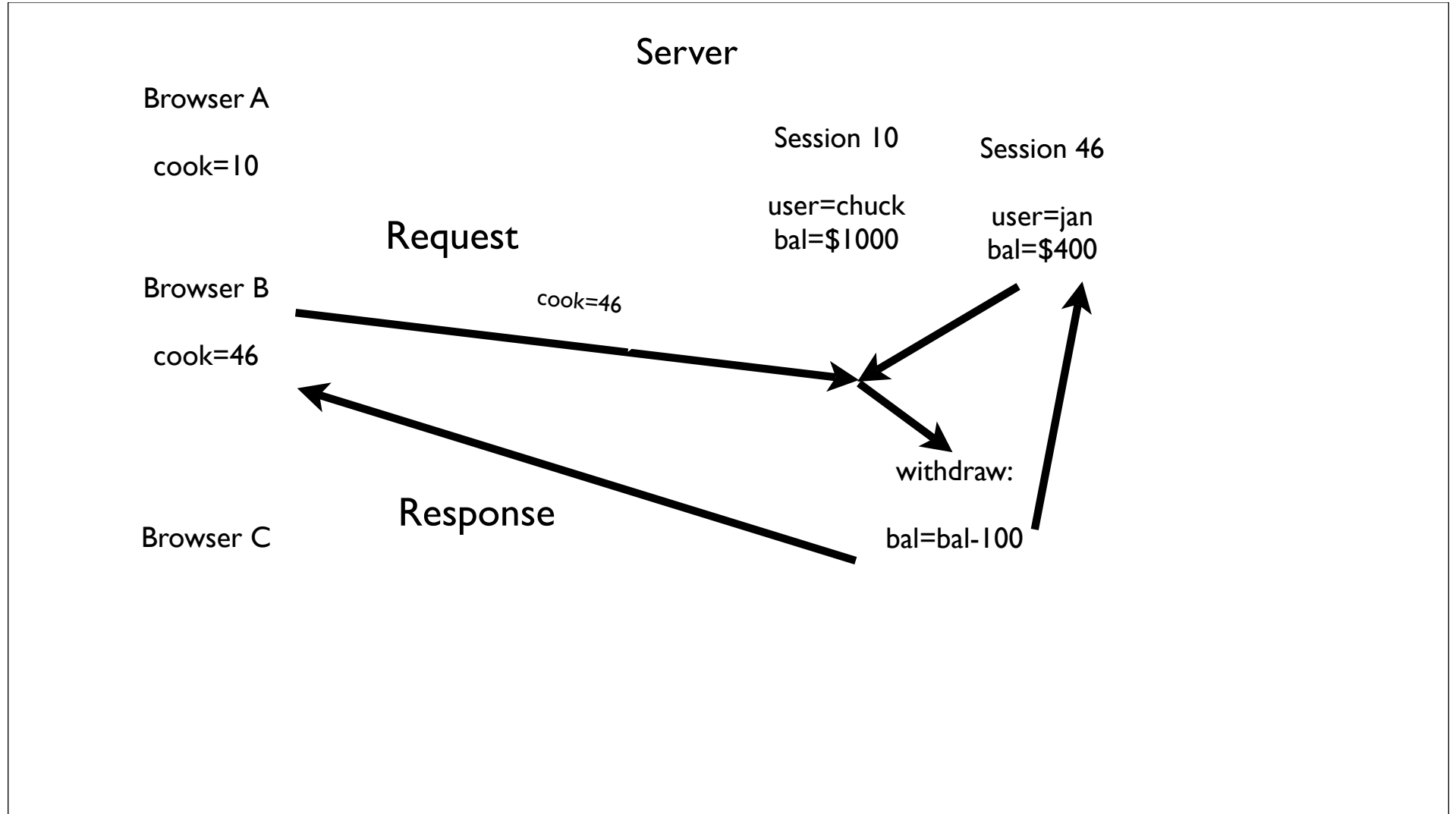
cook=46

withdraw:

bal=bal-100

Response

Browser C



Server

Browser A

cook=10

Browser B

cook=46

Browser C

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400



Click

Server

Browser A

cook=10

Browser B

cook=46

Browser C

cook=97

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

Session 97

Create
Session

Request

cook=97

Response

index:

"Please
log in"

Server

Browser A

cook=10

Browser B

cook=46

Browser C

cook=97

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

Session 97

Typing

Server

Browser A

cook=10

Browser B

cook=46

Browser C

cook=97

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

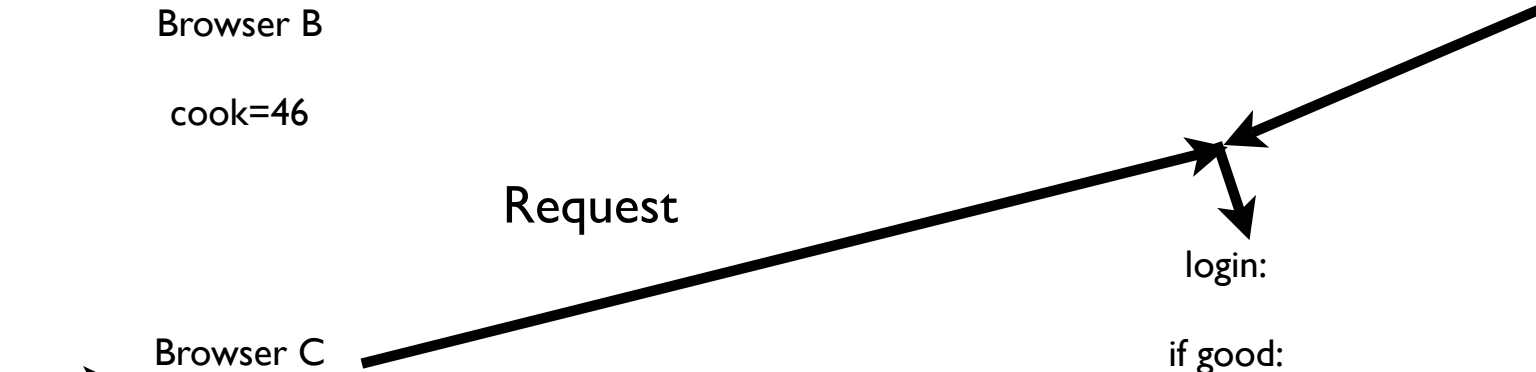
Session 97

Request

login:

if good:
set user

Click



Server

Browser A

cook=10

Browser B

cook=46

Browser C

cook=97

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

Session 97

user=phil

Request

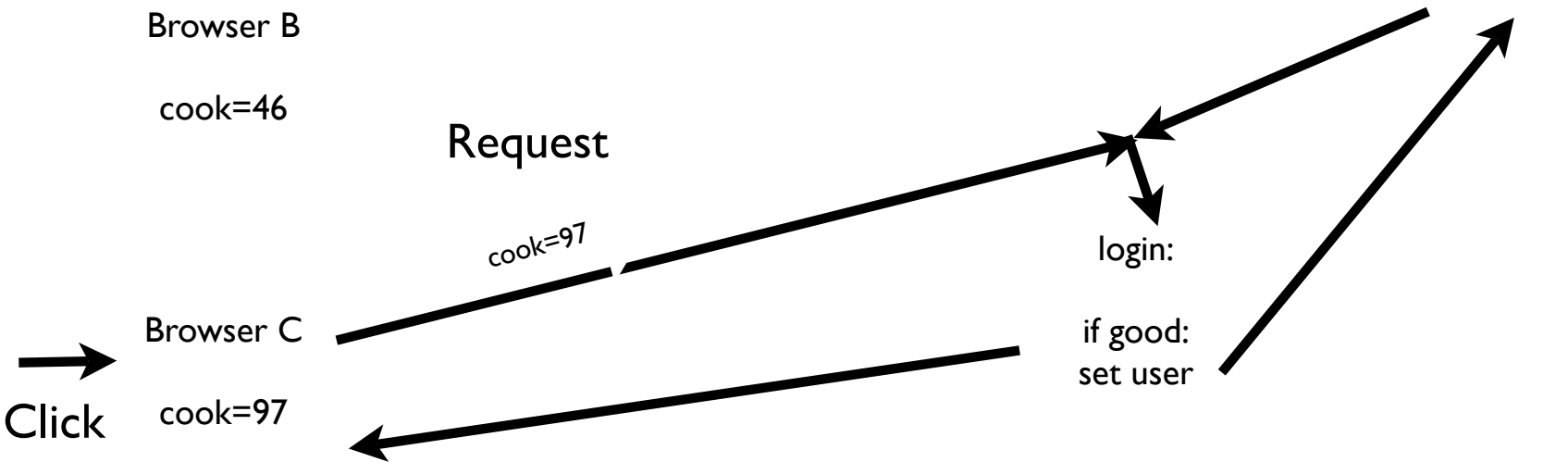
cook=97

login:

if good:
set user

Response

Click



Server

Browser A

cook=10

Browser B

cook=46

Browser C

cook=97

Session 10

user=chuck
bal=\$1000

Session 46

user=jan
bal=\$400

Session 97

user=phil

Using the Session in Rails

Session Variable

- In our view and controller there s a special hash map called “session” that persists across multiple HTTP Request / Response Cycles
- We can use this as a place to store long-lasting information such as the name of the current logged in user
- It is like flash - but it lasts across request-response cycles

Log messages

- We have been seeing session information all along from our very first Rails program in the logs
- The session ID changes if you close and re-open a browser or open two browsers (Safari and Firefox) at the same time

Processing Assn6Controller#index (for 127.0.0.1 at 2007-10-09 10:55:47) [GET]

Session ID: f012e17e4fe1d4c240a0cb34c69b2ab0

Parameters: {"action"=>"index", "controller"=>"assn6"}

Rendering assn6/index

Completed in 0.00195 (512 reqs/sec) | Rendering: 0.00096 (49%) | 200 OK [http://localhost/Assn6/]

Welcome to SI539

- **About**
- **Contact**
- **Pictures**
- **Membership**
- **Chat**
- **Application**
- **Logout**

Welcome to the SI539 Sakai

Here are two quotes from Ghandi that look like they reach the same conclusion.

Power is of two kinds. One is obtained by the use of force, the other by acts of love. *Power* based on force is effective and permanent then the other.

I hope to demonstrate that the real source of authority by a few, but by the acquisition of authority, when abused.

Cookie Information - http://localhost:3000/One

Sakai dr-chuck.com NWA Chuck's Media iPhone Navigation I3k CT SI 182 Sakai Based Service : ...

Disable Cookies Clear Session Cookies Delete Domain Cookies Delete Path Cookies View Cookie Information Add Cookie...

http://localhost:3000/One

1 cookie

NAME	VALUE	HOST	PATH	SECURE	EXPIRES
_assn6_session_id	fa3e49c915aad8bd9af1f768b8081aa6	localhost	/	No	At End Of Session

Done

Using the Session Hash

- To remove an entry, simply set the session to be nil

```
session[:lasalle] = params[:account]
```

```
if session[:lasalle] != nil
```

```
end
```

```
session[:lasalle] = nil
```

Objects in Session

- The session hash can store any object - not just strings

```
s = Story.new  
session[:currentstory] = s
```

- However we generally do not want to fill session up with too much “stuff” - we ust put in things like logged-in user name, current course, and things that allow us to “look up” other important things.

Sessions

Sessions allows you to store objects in between requests. This is useful for objects that are not yet ready to be persisted, such as a Signup object constructed in a multi-paged process, or objects that don't change much and are needed all the time, such as a User object for a system that requires login. The session should not be used, however, as a cache for objects where it's likely they could be changed unknowingly. It's usually too much work to keep it all synchronized — something databases already excel at.

You can place objects in the session by using the `session` method, which accesses a hash:

```
session[:person] = Person.authenticate(user_name, password)
```

And retrieved again through the same hash:

```
Hello #{session[:person]}
```

For removing objects from the session, you can either assign a single key to nil, like `session[:person] = nil`, or you can remove the entire session with `reset_session`.

By default, sessions are stored on the file system in `RAILS_ROOT/tmp/sessions`. Any object can be placed in the session (as long as it can be Marshalled). But remember that 1000 active sessions each storing a 50kb object could lead to a 50MB store on the filesystem. In other words, think carefully about size and caching before resorting to the use of the session on the filesystem.

<http://api.rubyonrails.org/classes/ActionController/Base.html>

Login / Logout Pattern

Login / Logout

- Having a session is not the same as being logged in.
- Generally you have a session the instant you connect to a web site
- The Session ID cookie is set when the first page is delivered
- Login puts user information in the session (stored in the server)
- Logout removes user information from the session

In Rails...

- We will pick a key to be where we store our logged in user object in the session.
- Login sets the entry and logout clears the entry
- The entry starts out empty when a brand new session is created when a browser first connects - so you are “not logged in” (*)

Some applications like Twitter and Flickr automate the login process by setting a long-term cookie in the browser - if this long-term cookie is present - you get auto-logged in.

Welcome to SI539

- About
- Contact
- Pictures
- Membership
- Chat
- Application

Please Log In

Required Information

Enter your E-Mail:

Enter your Password:

If you have lost your password, please send an E-Mail to membership@si539.com to have your password reset.

```
def login
  session[:lasalle] = nil
  if not request.post?
    return
  end

  if params[:yourpw] == nil or params[:yourmail] == nil or
    params[:yourpw] == "" or params[:yourmail] == ""
    flash[:notice] = "Please specify both E-Mail and password"
    return
  end
  memb = Member.find_by_email(params[:yourmail])
  logger.info "Retrieved member $#{memb}"
  if memb == nil or params[:yourpw] != 'secret'
    flash[:notice] = "Account / Password combination not found"
    return
  end
  session[:lasalle] = memb
  logger.info "User logged in:#{memb.email}"
  redirect_to :action => 'index'
end
```

Clear user information
in session.

Check for bad data
from the form.

Look up to see if the
id/password is right.

Store the member
object in the session
hash.

```
def login
  session[:lasalle] = nil
  if not request.post?
    return
  end

  if params[:yourpw] == nil or params[:yourmail] == nil or
    params[:yourpw] == "" or params[:yourmail] == ""
    flash[:notice] = "Please specify both E-Mail and password"
    return
  end
  memb = Member.find_by_email(params[:yourmail])
  logger.info "Retrieved member $#{memb}"
  if memb == nil or params[:yourpw] != 'secret'
    flash[:notice] = "Account / Password combination not found"
    return
  end
  session[:lasalle] = memb
  logger.info "User logged in:#{memb.email}"
  redirect_to :action => 'index'
end
```

Using Session Information

```
<h2>Real-Time Chat</h2>
```

```
<p>
```

```
<% if session[:lasalle] != nil %>
```

```
  <% form_remote_tag :url => 'chatcontent', :update => 'chatdiv' do -%>
```

```
    <input type="text" size="60" name="chatmsg"/>
```

```
    <%= submit_tag 'Send' %>
```

```
  <% end %>
```

```
<% else %>
```

```
  You must log in to participate in chat. (<%= link_to "login", :action => 'login' %>)
```

```
<% end %>
```

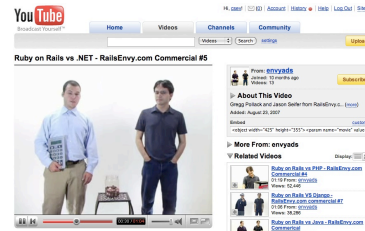
```
</p>
```

Logout is easy...

```
def logout
  session[:lasalle] = nil
  redirect_to :action => 'index'
end
```

Remove the user data
from the session hash.

Browser



Click Draw

Whole
Page

GET

You watch the YouTube video
for an 30 seconds

Click Draw

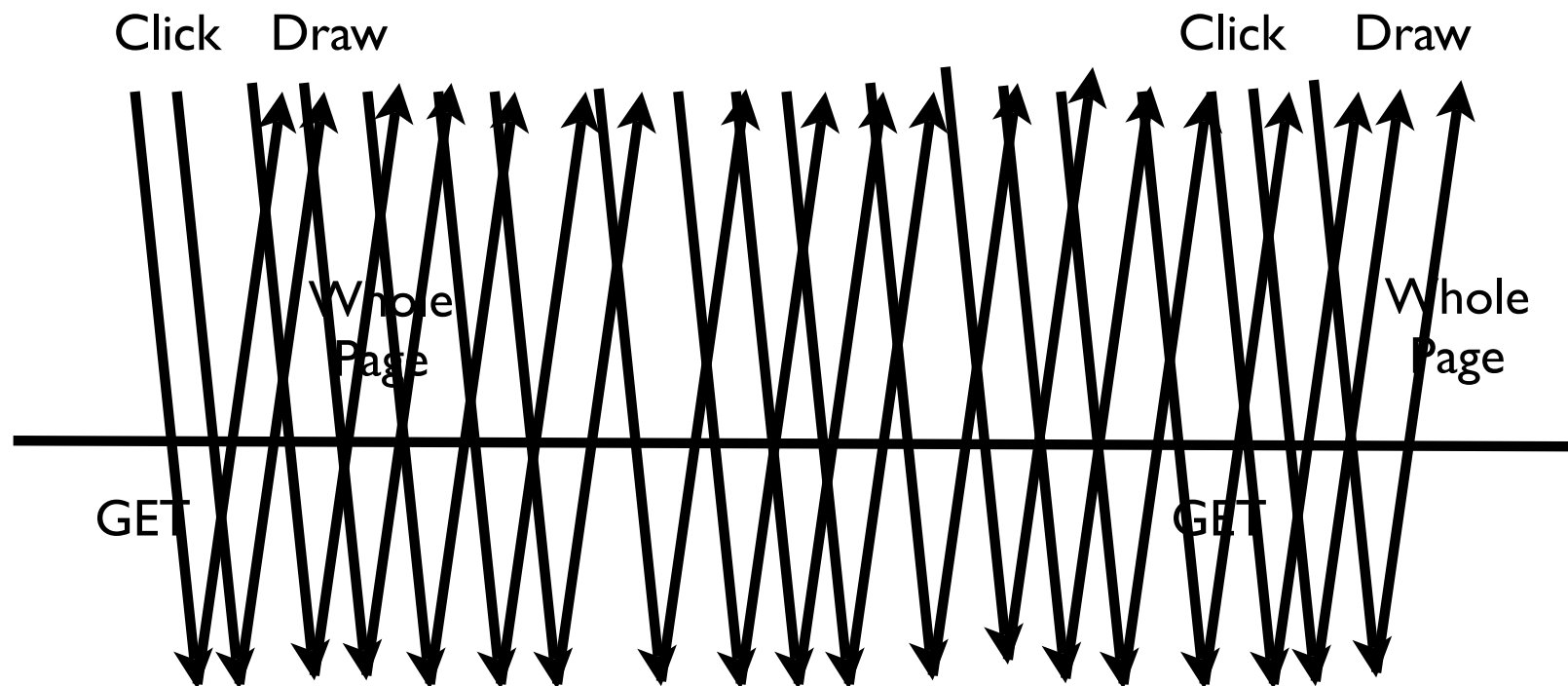
Whole
Page

GET

Server

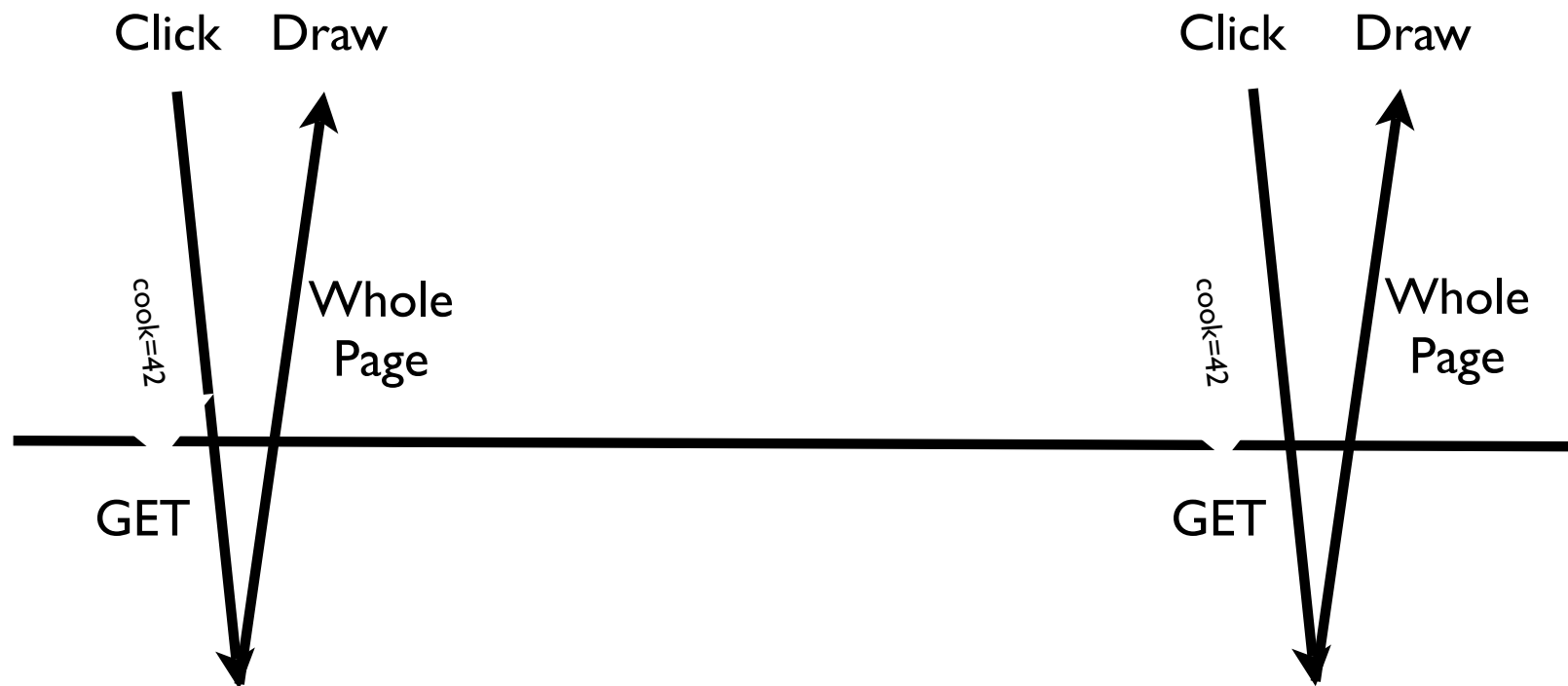
How you see YouTube...

Browser



Server

Browser



Server

Session 42

Session 42

Summary

- Cookies take the stateless web and allow servers to store small “breadcrumbs” in each browser.
- Session IDs are large random numbers stored in a cookie and used to maintain a session on the server for each of the browsers connecting to the server
- Rails applications can use the `session[]` hash map to store data across multiple request/response cycles.